Blazor EventCallback

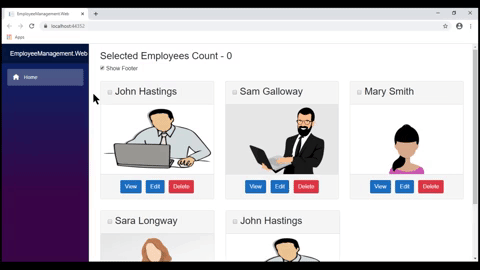
In this video we will discuss **how to communicate from child component to parent component using EventCallBacks** with an example.

On this page we have 2 blazor components

1. EmployeeList Component (Parent Component)
2. DisplayEmployee Component (Child Component)

In our previous video, we discussed how to pass data from parent component to child component using parameters. In this video, we will discuss how to do the opposite i.e pass data from child component to parent component. Let's understand this with an example.

Child to parent component communication



* As you can see from the image above, when an emmployee is selected by checking the checkbox, we want to display the Selected Employees Count.
* The Checkbox is in the child component (DisplayEmployee Component)
* The Selected Employees Count must be displayed in the parent component (EmployeeList Component)
* When the checkbox checked status changes in the child component, the selected employees count must be updated accordingly in the parent component. This means, we want to call the parent component method in response to the child component event.

Expose an event using EventCallback

The parent component must be notified, when the checkbox checked status changes in the child component. For this the child component exposes an event. The parent component assigns a callback method to the child component's event. In Blazor, to expose an event we use EventCallback.

Child Component Class (DisplayEmployeeBase.cs)

* OnEmployeeSelection is the event this child component is exposing. It is of type EventCallback.
* EventCallback supports generics which we use to pass event data (also commonly called event payload).
* In our example, we are passing bool. A value of true or false depending on whether the checkbox is checked or not.
* Checkbox binds to IsSelected boolean property.
* CheckBoxChanged method is the event hanlder. It stores the checked state of the checkbox in IsSelected property. It also raises OnEmployeeSelection custom event.
* Checkbox checked state is passed as the event payload to OnEmployeeSelection event.

public class DisplayEmployeeBase : ComponentBase

{

protected bool IsSelected { get; set; }

[Parameter]

public EventCallback<bool> OnEmployeeSelection { get; set; }

protected async Task CheckBoxChanged(ChangeEventArgs e)

{

IsSelected = (bool)e.Value;

await OnEmployeeSelection.InvokeAsync(IsSelected);

}

}

Copy

Child Component View (DisplayEmployee.razor)

In the view, the checkbox is bound to IsSelected property and CheckBoxChanged() is specified as the event handler for onchange event. Remember change event is fired every time the checked state of the checkbox changes.

<input type="checkbox" checked="@IsSelected" @onchange="CheckBoxChanged" />

Copy

Parent Component Class (EmployeeListBase.cs)

* SelectedEmployeesCount property keeps track of the number of employees selected. This is the property that we increment of decrement depending on the checked state of the checkbox in the child component.
* EmployeeSelectionChanged is the callback method. This method will be called when the checked state of the checkbox in the child component changes.

public class EmployeeListBase : ComponentBase

{

protected int SelectedEmployeesCount { get; set; } = 0;

protected void EmployeeSelectionChanged(bool isSelected)

{

if(isSelected)

{

SelectedEmployeesCount++;

}

else

{

SelectedEmployeesCount--;

}

}

}

Copy

Parent Component View (EmployeeList.razor)

In the view assign EmployeeSelectionChanged as the Callback method to the child component custom event OnEmployeeSelection.

@foreach (var employee in Employees)

{

<DisplayEmployee OnEmployeeSelection="EmployeeSelectionChanged">

</DisplayEmployee>

}